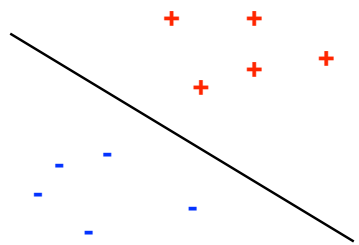
10-701 Introduction to Machine Learning (PhD)
Lecture 12: Boosting

Leila Wehbe
Carnegie Mellon University
Machine Learning Department

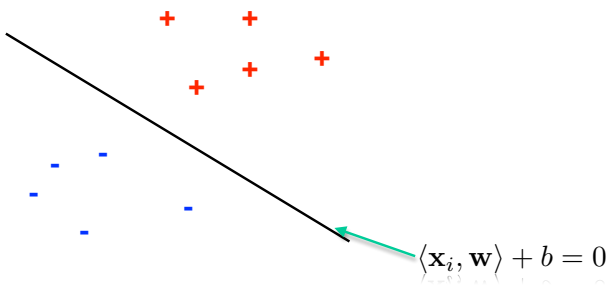
Slides based on Tom Mitchell's 10701 Fall 2016 material
Readings: Andrew Ng's lecture notes at:
http://cs229.stanford.edu/notes/cs229-notes3.pdf
and Rob Shapire's paper: http://www.cs.cmu.edu/~tom/10701-s16/schapire02boosting.pdf

SVMs

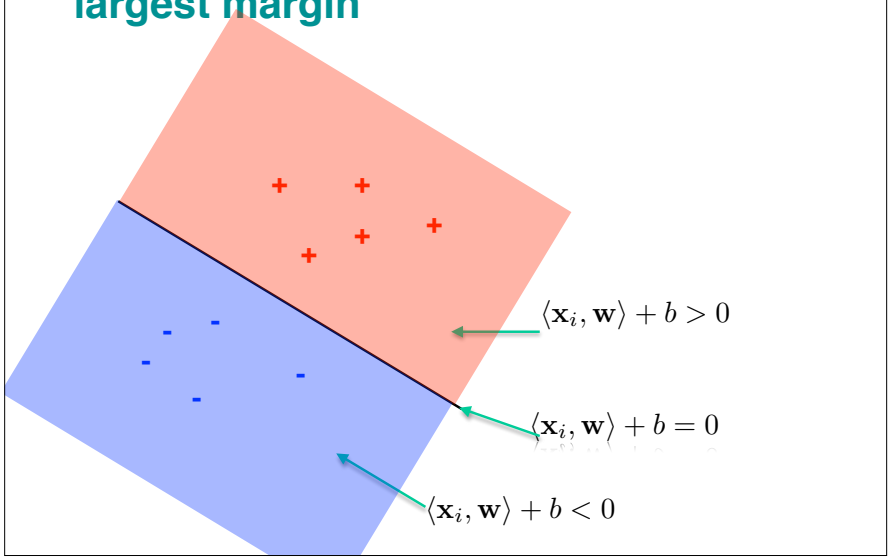
Find a linear separator with the largest margin



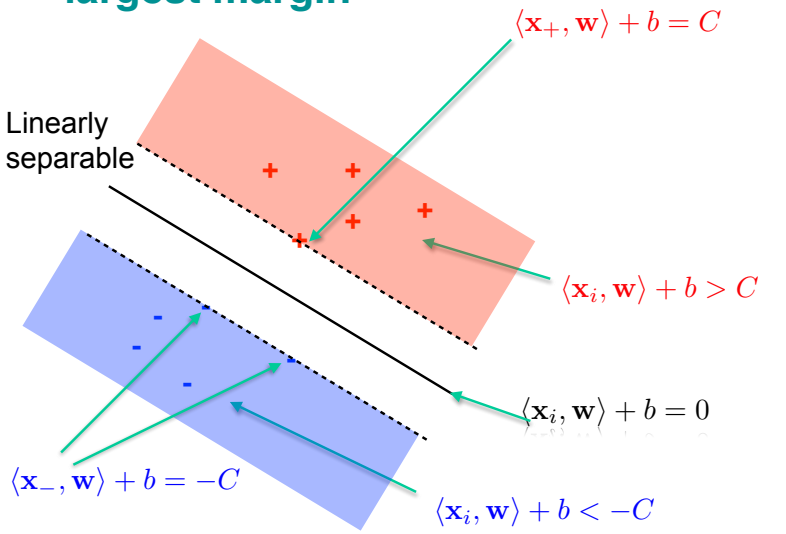
Find a linear separator with the largest margin



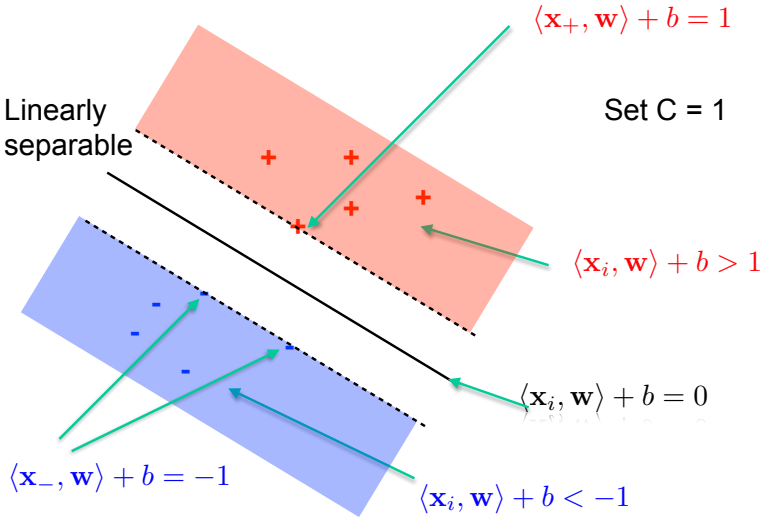
Find a linear separator with the largest margin



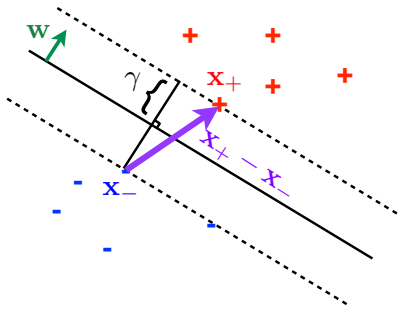
Find a linear separator with the largest margin



Find a linear separator with the largest margin



Find a linear separator with the largest margin

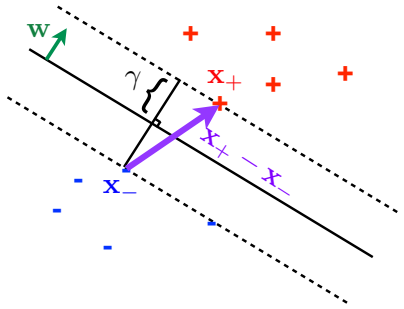


$$2\gamma = \frac{\langle \mathbf{x}_+ - \mathbf{x}_-, \mathbf{w} \rangle}{||\mathbf{w}||}$$

$$\gamma = \frac{1}{2} \frac{\langle \mathbf{x}_+ - \mathbf{x}_-, \mathbf{w} \rangle}{||\mathbf{w}||}$$

Find a linear separator with the largest margin

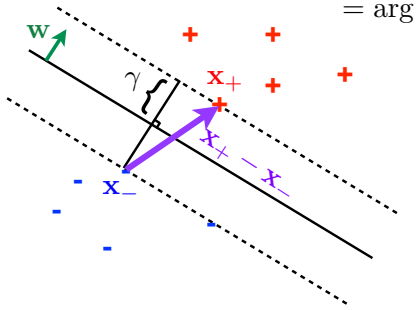
$$\arg\max_{\mathbf{w},b} \gamma = \arg\max_{\mathbf{w},b} \frac{1}{2} \frac{\langle \mathbf{x}_+ - \mathbf{x}_-, \mathbf{w} \rangle}{||\mathbf{w}||}$$



Find a linear separator with the largest margin

$$\arg\max_{\mathbf{w},b} \gamma = \arg\max_{\mathbf{w},b} \frac{1}{2} \frac{\langle \mathbf{x}_+ - \mathbf{x}_-, \mathbf{w} \rangle}{||\mathbf{w}||}$$

$$= \arg\max_{\mathbf{w},b} \frac{1}{2} \frac{\langle \mathbf{x}_+, \mathbf{w} \rangle - \langle \mathbf{x}_-, \mathbf{w} \rangle}{||\mathbf{w}||}$$



$\langle \mathbf{x}_+, \mathbf{w} \rangle + b = 1$

$\langle \mathbf{x}_-, \mathbf{w} \rangle + b = -1$

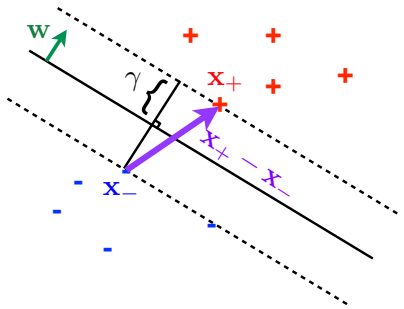
Find a linear separator with the largest margin

$$\arg\max_{\mathbf{w},b} \gamma = \arg\max_{\mathbf{w},b} \frac{1}{2} \frac{1 - b - (-1 - b)}{||\mathbf{w}||}$$

$$= \arg\max_{\mathbf{w},b} \frac{1}{2} \frac{2}{||\mathbf{w}||} = \arg\max_{\mathbf{w},b} \frac{1}{||\mathbf{w}||}$$

$$= \arg\min_{\mathbf{w},b} ||\mathbf{w}||$$

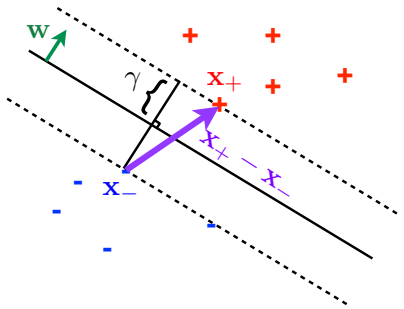
$$= \arg\min_{\mathbf{w},b} \frac{1}{2} ||\mathbf{w}||^2$$



The (primal) optimization problem is:

$$\min_{\mathbf{w},b} \frac{1}{2} ||\mathbf{w}||^2$$

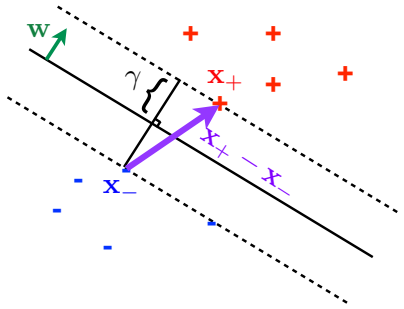
$$\text{s.t.} \quad y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad i = 1, ..., m$$



The (primal) optimization problem is:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t.} \quad y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad i = 1, ..., m$$



This can be written as:

$$\min_{\mathbf{w},b} \; f(\mathbf{w}, b)$$

$$\text{s.t.} \quad g(\mathbf{w}, b) \leq 0, \quad i = 1, ..., m$$

Where $f(\mathbf{w}, b) = ||\mathbf{w}||^2$ and $g(\mathbf{w}, b) = 1 - y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b)$ are both convex functions of our parameters **w** and b

Lagrangian

We can write the Lagrangian of:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t.} \quad y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad i = 1, ..., m$$

as: $$\mathcal{L}(\mathbf{w}, b, \alpha) = f(\mathbf{w}, b) + \sum_{i=1}^{m} \alpha_i g_i(\mathbf{w}, b)$$

$$= \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i \left(1 - y_i(\langle \mathbf{x_i}, \mathbf{w} \rangle + b)\right)$$

$$g_i(\mathbf{w}, b) = 1 - y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b)$$

Lagrangian

We can write the Lagrangian of:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t.} \quad y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad i = 1, ..., m$$

as: $$\mathcal{L}(\mathbf{w}, b, \alpha) = f(\mathbf{w}, b) + \sum_{i=1}^{m} \alpha_i g_i(\mathbf{w}, b)$$

$$= \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i \left(1 - y_i(\langle \mathbf{x_i}, \mathbf{w} \rangle + b)\right)$$

Take the quantity:

$$\theta_{\mathcal{P}}(\mathbf{w}, b) = \max_{\alpha, \alpha_i \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha)$$

$$\theta_{\mathcal{P}}(\mathbf{w}, b) = \begin{cases} f(\mathbf{w}, b) & \text{if } \mathbf{w}, b \text{ satisfy primal constraint} \\ \infty & \text{otherwise} \end{cases}$$

Primal and dual

$$\theta_{\mathcal{P}}(\mathbf{w}, b) = \max_{\alpha, \alpha_i \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha)$$

$$\theta_{\mathcal{P}}(\mathbf{w}, b) = \begin{cases} f(\mathbf{w}, b) & \text{if } \mathbf{w}, b \text{ satisfy primal constraint} \\ \infty & \text{otherwise} \end{cases}$$

The solution to:

$$\min_{\mathbf{w},b} \theta_{\mathcal{P}}(\mathbf{w}, b) = \min_{\mathbf{w},b} \max_{\alpha, \alpha_i \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha)$$

This is the same problem as our original primal problem and therefore has the same solution.
Original problem:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t.} \quad y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad i = 1, ..., m$$

Primal and dual

Now take a slightly different problem:

$$\theta_{\mathcal{D}}(\alpha) = \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha)$$

The dual optimization problem is:

$$\max_{\alpha,\alpha_i \geq 0} \theta_{\mathcal{D}}(\alpha) = \max_{\alpha,\alpha_i \geq 0} \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha)$$

We have that:

$$d^* = \max_{\alpha,\alpha_i \geq 0} \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) \leq \min_{\mathbf{w},b} \max_{\alpha,\alpha_i \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha) = p^*$$

Under certain conditions (next slide): $d^* = p^*$

Primal and dual

Both satisfied in separable SVM formulation

If:
- f and g are convex
- There exist w that satisfies constraint g (i.e. there exists a w for which g(w)<0)

then:
- there exists **w*** and b* solutions to the primal problem, and α^* solution to the dual problem, and:

$$d^* = p^* = \mathcal{L}(\mathbf{w}^*, b^*, \alpha^*)$$

Also, $\mathbf{w}^*, b^*, \alpha^*$ satisfy the KKT conditions:

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}, b^*, \alpha^*)}{\partial w_i}\right|_{\mathbf{w}^*} = 0, \ i = 1, ..., n$$

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}^*, b, \alpha^*)}{\partial b}\right|_{b^*} = 0$$

$$\alpha_i^* g_i(\mathbf{w}^*) = 0, \ i = 1, ..., m$$

$$g_i(\mathbf{w}^*) \leq 0, \ i = 1, ..., m$$

$$\alpha_i \geq 0, \ i = 1, ..., m$$

Primal and dual

Both satisfied in separable SVM formulation

If:
- f and g are convex
- There exist w that satisfies constraint g (i.e. there exists a w for which g(w)<0)

then:
- there exists **w*** and b* solutions to the primal problem, and α^* solution to the dual problem, and:

$$d^* = p^* = \mathcal{L}(\mathbf{w}^*, b^*, \alpha^*)$$

Also, $\mathbf{w}^*, b^*, \alpha^*$ satisfy the KKT conditions:

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}, b^*, \alpha^*)}{\partial w_i}\right|_{\mathbf{w}^*} = 0, \ i = 1, ..., n$$

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}^*, b, \alpha^*)}{\partial b}\right|_{b^*} = 0$$ Stationarity

Complementary slackness

$$\alpha_i^* g_i(\mathbf{w}^*) = 0, \ i = 1, ..., m$$

$$g_i(\mathbf{w}^*) \leq 0, \ i = 1, ..., m$$

Feasibility $\alpha_i \geq 0, \ i = 1, ..., m$

KKT conditions

Stationarity: **w***,b* local extremum of Lagrangian for fixed α^*

Feasibility: All primal and dual constraints are satisfied

Complementary Slackness: either $\alpha_i = 0$ or $g_i(\mathbf{w}, b) = 0$

If $\alpha_i > 0$ then $g_i(\mathbf{w}, b) = 0$

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}, b^*, \alpha^*)}{\partial w_i}\right|_{\mathbf{w}^*} = 0, \ i = 1, ..., n$$

$$\left.\frac{\partial \mathcal{L}(\mathbf{w}^*, b, \alpha^*)}{\partial b}\right|_{b^*} = 0$$ Stationarity

Complementary slackness

$$\alpha_i^* g_i(\mathbf{w}^*) = 0, \ i = 1, ..., m$$

$$g_i(\mathbf{w}^*) \leq 0, \ i = 1, ..., m$$

Feasibility $\alpha_i \geq 0, \ i = 1, ..., m$

Solve the dual problem. First solve:

$$\min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \min_{\mathbf{w},b} f(\mathbf{w}, b) + \sum_{i=1}^{m} \alpha_i g_i(\mathbf{w}, b)$$

$$= \min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i \left(1 - y_i(\langle \mathbf{x_i}, \mathbf{w} \rangle + b)\right)$$

$$s.t. \quad \alpha \geq 0$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha)}{\partial b} = 0$$

Solving the dual problem. First solve:

$$\min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \min_{\mathbf{w},b} f(\mathbf{w}, b) + \sum_{i=1}^{m} \alpha_i g_i(\mathbf{w}, b)$$

$$= \min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i \left(1 - y_i(\langle \mathbf{x_i}, \mathbf{w} \rangle + b)\right)$$

$$s.t. \quad \alpha \geq 0$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0 \qquad \mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \alpha)}{\partial b} = 0 \qquad \sum_i \alpha_i y_i = 0$$

Then replace:

$$\min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

Then solve:

$$\max_{\alpha, \alpha_i \geq 0} \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \max_{\alpha, \alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$s.t. \quad \alpha_i \geq 0$$

$$\sum \alpha_i y_i = 0$$

After solving for α :

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = 1 - \langle \mathbf{x}_+, \mathbf{w} \rangle$$

Then solve:

$$\max_{\alpha,\alpha_i \geq 0} \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \max_{\alpha,\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$s.t. \quad \alpha_i \geq 0$$

$$\sum \alpha_i y_i = 0$$

After solving for α :

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = 1 - \langle \mathbf{x}_+, \mathbf{w} \rangle$$

For new points, predict:

$$\text{sign}\left(\langle \mathbf{w}, \mathbf{x}_j \rangle + b\right) = \text{sign}\left(\left\langle \sum_i \alpha_i y_i \mathbf{x}_i, \mathbf{x}_j \right\rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b\right)$$

Then solve:

$$\max_{\alpha,\alpha_i \geq 0} \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \alpha) = \max_{\alpha,\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$s.t. \quad \alpha_i \geq 0$$

$$\sum \alpha_i y_i = 0$$

After solving for α :

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = 1 - \langle \mathbf{x}_+, \mathbf{w} \rangle$$

For new points, predict:

$$\text{sign}\left(\langle \mathbf{w}, \mathbf{x}_j \rangle + b\right) = \text{sign}\left(\left\langle \sum_i \alpha_i y_i \mathbf{x}_i, \mathbf{x}_j \right\rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b\right)$$

Kernels

A kernel is a function of two variables that can be written as dot product of the same feature map of these variables:

$$K(x, z) = \phi(x)^T \phi(z).$$

Theorem (Mercer). Let $K : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ be given. Then for K to be a valid (Mercer) kernel, it is necessary and sufficient that for any $\{x^{(1)}, \ldots, x^{(m)}\}$, $(m < \infty)$, the corresponding kernel matrix is symmetric positive semi-definite.

Example kernel

$$\phi(x) = \begin{bmatrix} x \\ x^2 \\ x^3 \end{bmatrix}$$

$$k(x, y) = \langle \phi(x), \phi(y) \rangle = xy + x^2 y^2 + x^3 y^3$$

Example kernel

$$K(x,z) = (x^T z)^2.$$

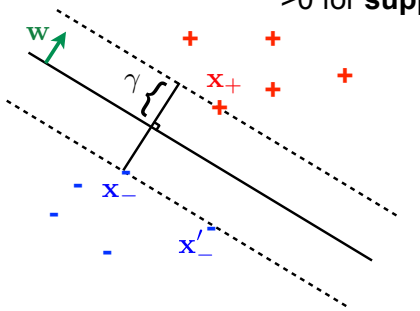
$$\begin{aligned} K(x,z) &= \left(\sum_{i=1}^{n} x_i z_i\right)\left(\sum_{j=1}^{n} x_j z_j\right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j z_i z_j \\ &= \sum_{i,j=1}^{n} (x_i x_j)(z_i z_j) \end{aligned} \qquad \phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix}.$$

For new points:

For new $\mathbf{x}_j$, classifier output is:

$$\text{sign}\left(\langle \mathbf{w}, \mathbf{x}_j \rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b\right)$$

>0 for **support vectors**



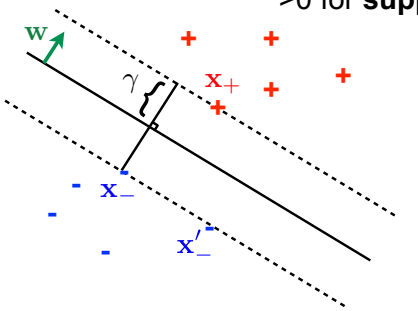
https://www.flickr.com/photos/30686429@N07/albums/72157622330082619/with/3953914089/

For new points:

For new $\mathbf{x}_j$, classifier output is:

$$\text{sign}\left(\langle \mathbf{w}, \mathbf{x}_j \rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b\right)$$

>0 for **support vectors**

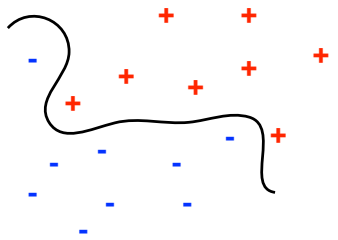


Captures similarity to support vectors: more similar support vectors have more weight

The kernel trick

$$\text{sign}\left(\sum_i \alpha_i y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_j) + b\right)$$

Avoids computing explicit feature maps

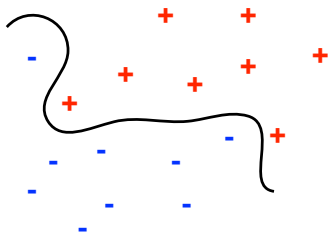


The feature map lifts the data points to a higher-dimensional space where they could be linearly separable.

This allows us to learn a non-linear decision boundary in the original space.

Learn non-linear boundaries

$$\text{sign}\left(\sum_i \alpha_i y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_j) + b\right)$$



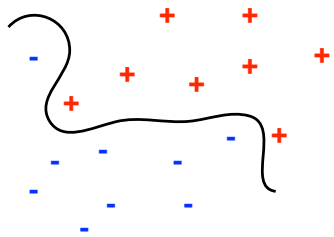
Example:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right)$$

Closer support vectors have more weight than far away ones when classifying $\mathbf{x}_j$

Learn non-linear boundaries

$$\text{sign}\left(\sum_i \alpha_i y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle + b\right) = \text{sign}\left(\sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_j) + b\right)$$

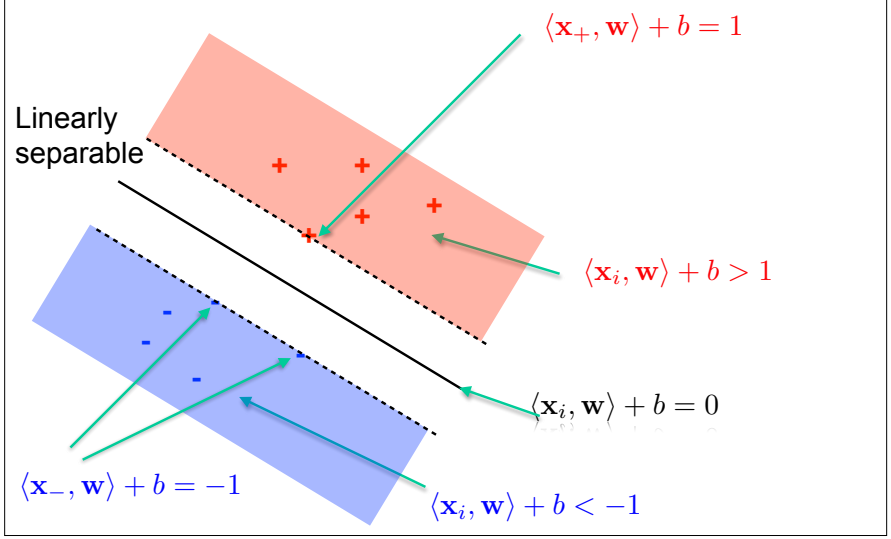


Example:

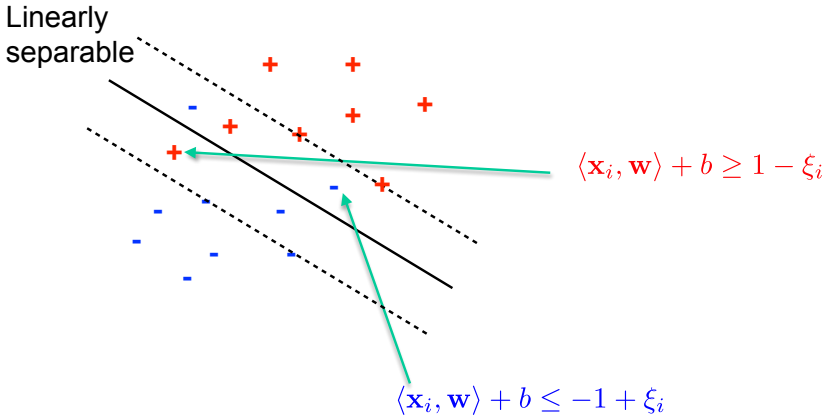
$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right)$$

More in HW3

Separable



Non separable



Non separable

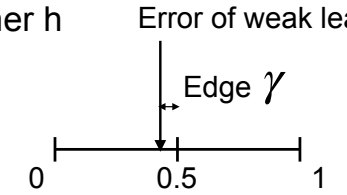
$$\min_{\gamma,w,b} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i$$
$$\text{s.t.} \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, \quad i = 1, \ldots, m$$
$$\xi_i \geq 0, \quad i = 1, \ldots, m.$$

More in HW3

Boosting

Boosting

- A weak learner h



- Use a combination of weak learners to obtain a strong classifier

$$H(x) = h_1(x) + h_2(x) + h_3(x) \ldots$$

Boosting: Key Idea

[Rob Shapire]

- Use a learner that produces better-than-chance h(x)'s
- Train it multiple times, on **reweighted training examples**
  - Each time, upweight the incorrectly classified examples, downweight the correctly classified examples
- Final prediction: weighted vote of the multiple $h_i(x)$'s

- Practically useful
- Theoretically interesting

$$H(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) \ldots$$

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

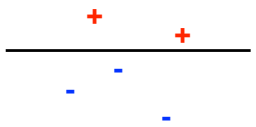
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

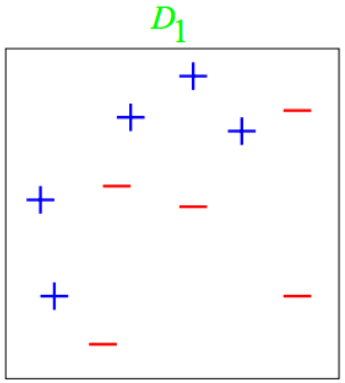
Can use any classifier!

- We will take here **decision stumps** as illustration, but AdaBoost can be used with any classifier



AdaBoost: A toy example

Weak classifiers: vertical or horizontal half-planes (a.k.a. decision stumps)

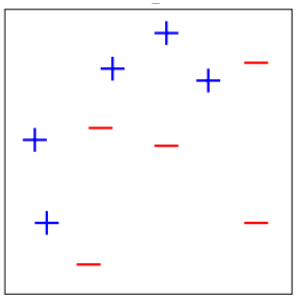


[Rob Shapire]

Quiz

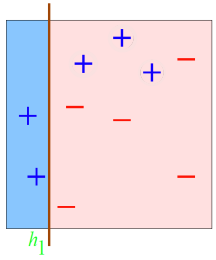
https://goo.gl/YY85mM

Q1- Initialization



What are the w_i^1 initialized to?

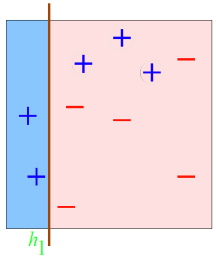
Q2- First iteration



This decision stump is chosen first.

What is the error ϵ_1 of h_1?

Q3- First iteration



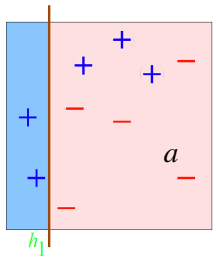
This decision stump is chosen first.

What is α_1?

A- $\frac{1}{2}\ln\left(\frac{0.7}{0.3}\right) \approx 0.42$

B- $\frac{1}{2}\ln\left(\frac{0.3}{0.7}\right) \approx -0.42$

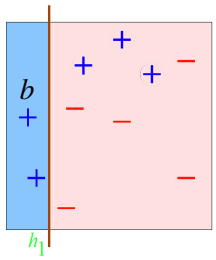
Q4- First iteration



This decision stump is chosen first.

Is $w_2^a > w_1^a$?

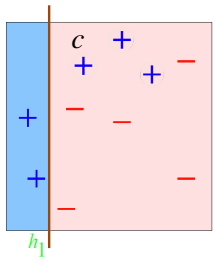
Q5- First iteration



This decision stump is chosen first.

Is $w_2^b > w_1^b$?

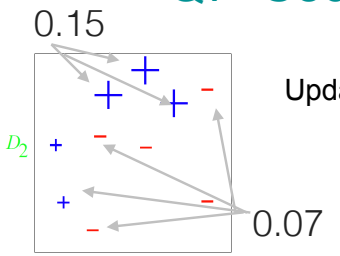
Q6- First iteration



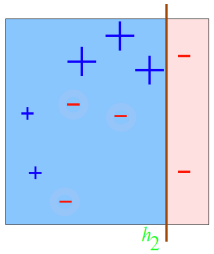
This decision stump is chosen first.

Is $w_2^c > w_1^c$?

Q7- Second iteration



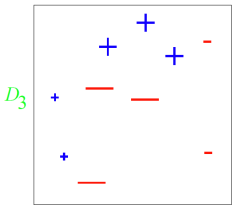
Updated dataset with weights.



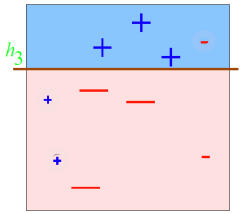
This decision stump is chosen second.

What is the error ϵ_2 of h_2?

Q8- Third iteration



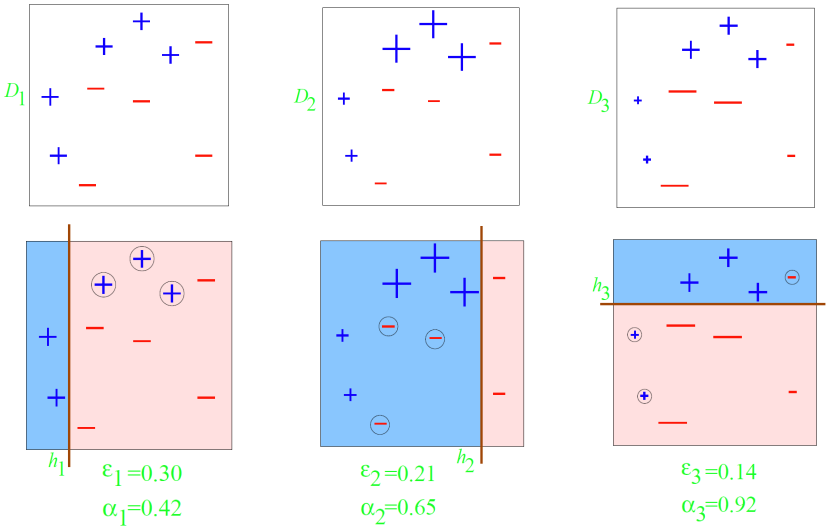
Updated dataset with weights.



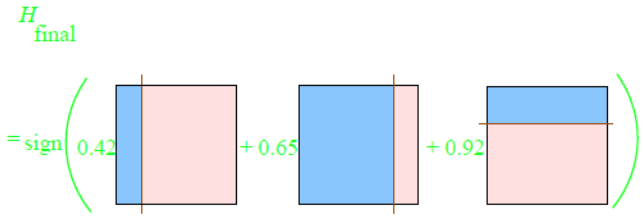
This decision stump is chosen third.

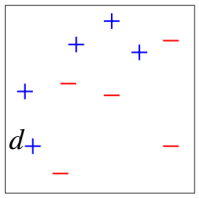
How many examples are misclassified by h_3 ?

Resulting weights



Q10- Classifier

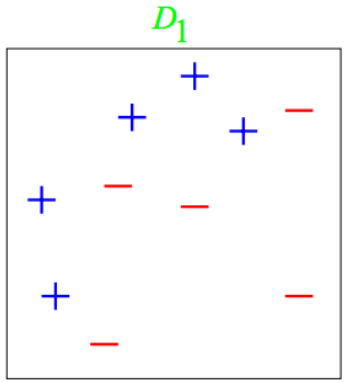




What is d classified as?
(blue is positive)

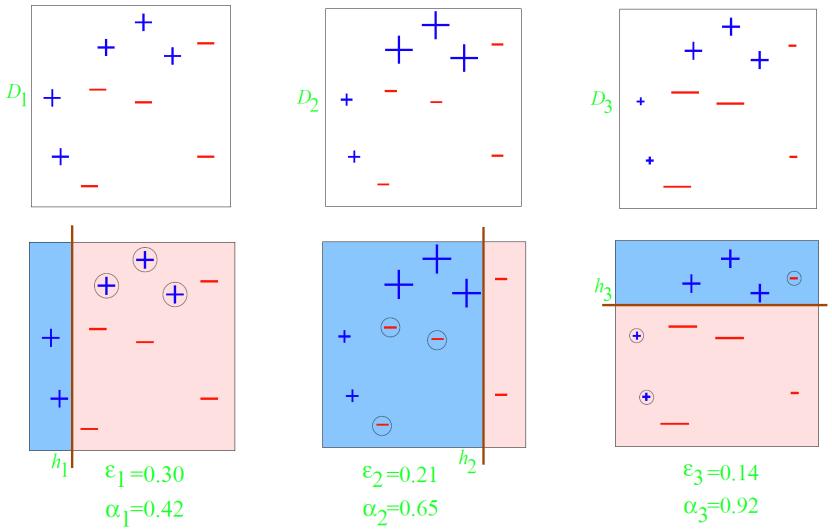
AdaBoost: A toy example

Weak classifiers: vertical or horizontal half-planes (a.k.a. decision stumps)

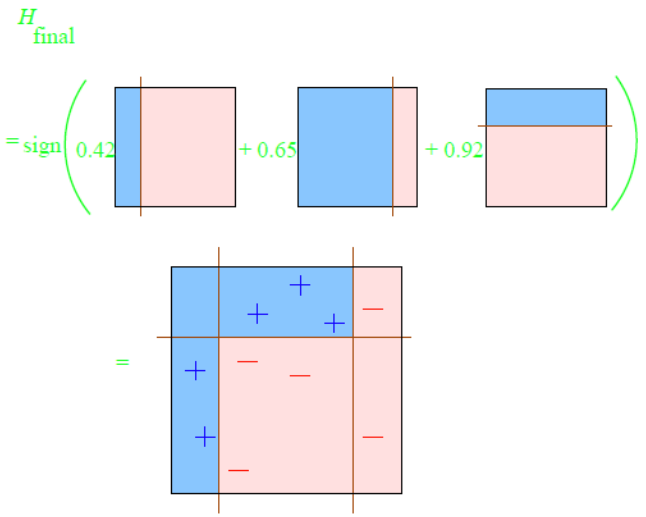


[Rob Shapire]

AdaBoost: A toy example



AdaBoost: A toy example



[Rob Shapire]

Could be any classifier

- Not just a decision stump
- AdaBoost is a meta-classifier

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Training Error

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i f(x_i))$$

Where $f(x) = \sum \alpha_t h_t(x); H(x) = sign(f(x))$

Theoretical Result 1: Training Error

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m} \delta(H(x_i) \neq y_i) \leq \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$

$$Z_t = \sum_{i=1}^{m} D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

Training error of final classifier is bounded by:

$$\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i)\neq y_i) \leq \frac{1}{m}\sum_{i}\exp(-y_i f(x_i)) = \prod_t Z_t$$

Where $f(x) = \sum_t \alpha_t h_t(x); H(x) = sign(f(x))$

If we minimize $\prod_t Z_t$, we minimize our training error

We can tighten this bound greedily, by choosing α_t and h_t on each iteration to minimize Z_t.

$$Z_t = \sum_{i=1}^{m} D_t(i)\exp(-\alpha_t y_i h_t(x_i))$$

Theoretical Result 1: Training Error

We can minimize this bound by choosing α_t on each iteration to minimize Z_t.

$$Z_t = \sum_{i=1}^{m} D_t(i)\exp(-\alpha_t y_i h_t(x_i))$$

For boolean target function, this is accomplished by [Freund & Schapire '97]:

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

Where:

$$\epsilon_t = \sum_{i=1}^{m} D_t(i)\delta(h_t(x_i)\neq y_i)$$

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i\sim D_t}\left[h_t(x_i)\neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t}\times\begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i)=y_i \\ e^{\alpha_t} & \text{if } h_t(x_i)\neq y_i \end{cases} \\ &= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right).$$

Analyzing the training error

- Theorem:
  - write ϵ_t as $1/2 - \gamma_t$
  - then

$$\begin{aligned} \text{training error}(H_{\text{final}}) &\leq \prod_t\left[2\sqrt{\epsilon_t(1-\epsilon_t)}\right] \\ &= \prod_t\sqrt{1-4\gamma_t^2} \\ &\leq \exp\left(-2\sum_t\gamma_t^2\right) \end{aligned}$$

- so: if $\forall t:\ \gamma_t \geq \gamma > 0$
  then $\text{training error}(H_{\text{final}}) \leq e^{-2\gamma^2 T}$
- AdaBoost is adaptive:
  - does not need to know γ or T a priori
  - can exploit $\gamma_t \gg \gamma$

[Rob Shapire]

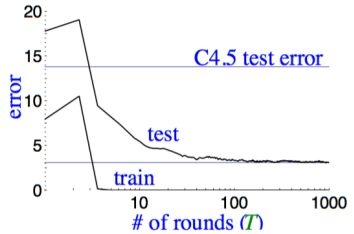
Bound on True Error

[Freund & Shapire, 1999]

With high probability:

$$error_{true}\left(sign(\sum_t \alpha_t h_t(x))\right) \leq error_{train}\left(sign(\sum_t \alpha_t h_t(x))\right) + O\left(\sqrt{\frac{T \cdot VCdim(H)}{m}}\right)$$

Actual Typical Run



(boosting C4.5 on "letter" dataset)

- test error does not increase, even after 1000 rounds
  - (total size > 2,000,000 nodes)
- test error continues to drop even after training error is zero!

	# rounds		
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1

[Rob Shapire]

AdaBoost and AdaBoost.MH on Train (left) and Test (right) data from Irvine repository. [Schapire and Singer, ML 1999]